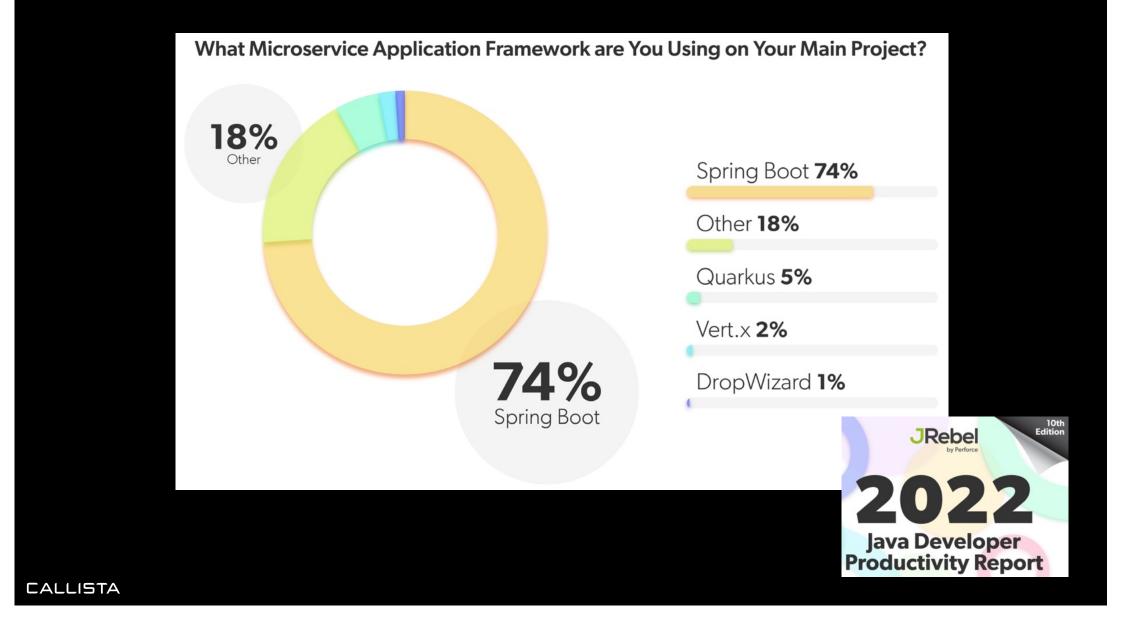
# SPRING BOOT 3 SPRING FRAMEWORK 6

MAGNUS LARSSON

CADEC 2023.01.19 & 2023.01.25 | CALLISTAENTERPRISE.SE





### AGENDA

- Overview
- Migration
- Native Compile
- Observability
- Summary



#### OVERVIEW

• The evolution of Java



## **New Requirements**

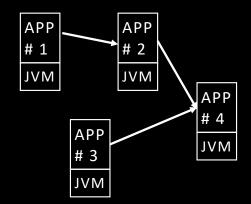
- Faster startup
  - Shorten warmup
- Less memory
- Scalability
  - Scale to zero
- Observability











**Distributed Systems** 

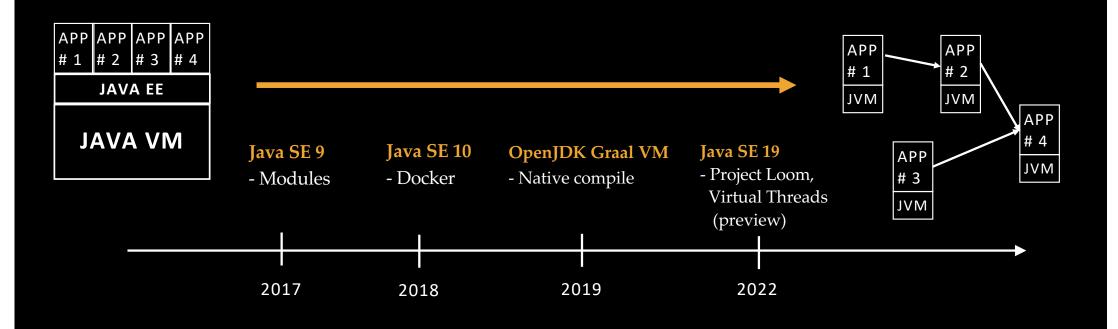


#### OVERVIEW

• The evolution of Java

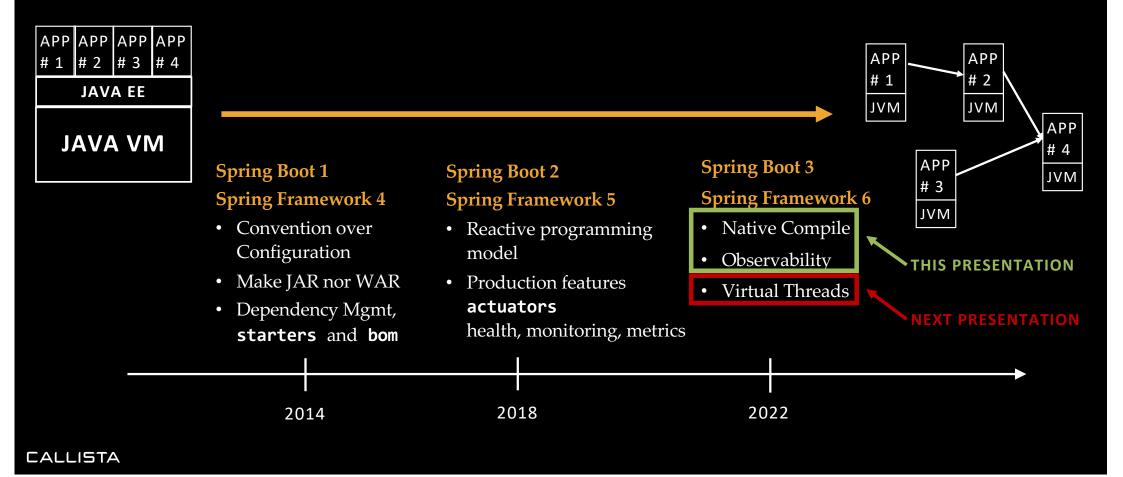
#### Emerging OpenJDK projects:

- CRaC, Amber, Valhalla, Leyden, and Panama



### OVERVIEW

• The evolution of **Spring** 



### AGENDA

- Overview
- Migration
- Native Compile
- Observability
- Summary



### MIGRATION

- Java 17 baseline
- Jakarta EE: Package rename: javax → jakarta
  - E.g. JPA: javax.persistence.\*  $\rightarrow$  jakarta.persistence.\*
- Deprecated code in 2.x removed

tasks.withType(JavaCompile) {
 options.compilerArgs += ['-Xlint:deprecation']

- Breaking changes etc
  - Spring Boot 3.0 Migration Guide
  - Spring Security 6.0 Migration Guide
- Importance of end-to-end black-box tests
  - Run them before and after the migration!

### AGENDA

- Overview
- Migration
- Native Compile
- Observability
- Summary



For details on GraalVM Native Image, see this presentation

Problem definition from Cadec 2021 - GraalVM Native Image

JVM based Micro Services

- 1. Large memory footprint
- 2. Long startup time
- 3. Initial warmup required (JIT)

Makes it expensive for large systems and impossible to scale to zero.



• Concerns from Cadec 2021

### Develop and Build Findings

- Unable to compile static executable and build from a scratch dockerimage
  - Both Go and GraalVM native executables depends on shared C/C++ libraries
  - Googles gcr.io/distroless/base is used instead
- Even minor changes breaks the build
  - Spring Boot 2.4.0-RC1 to 2.4.0 release update
  - Graal 20.2 to 20.3 minor update
  - Use of new features from existing 3rd party libraries
  - Adding 3rd party libraries
- Discrepancy between dev and runtime environments
- What's the credibility of unit tests
- Frameworks/libs without native support
- Use and maintain configurations for Reflection, Proxies, Resources and JNI

#### Road to Enable Native

#### GraalVM 20.3

- 1. Upgrade to Spring Boot 2.4
- 2. Add GraalVM native support. Substrate VM (svm)
- 3. Add Spring native support (spring-graalvm-native)
- 4. Create build script or use maven plugin (build.sh)
- 5. Declare all Reflections (for DTO beans) and resources
  Manually or use native-image-agent to generate
- 6. Compile, run and fix remaining stuff (trial and error)
  Reflection config for Kafka and ISON serializers
  - Resource config for Kafka
- Substitute Kafka class using Method Handles

#### **Requires Configuration**

- Reflections, Dynamic Class Loading
- Dynamic Proxies (JDK)
- Resource Access
- Java Native Interface (JNI)

native-image --initial
[total]: 687,593.47 ms

• Outcome from Cadec 2021



Dear fellow JVM'ers!

"There's no Holy Graal, just loads of hard work and Java."

• Is it better now?





- With Spring Boot 3 and Spring Framework 6
  - Compile Spring Boot applications into standalone executables, called a native image
  - Uses GraalVM native-image compiler
    - » New build module Spring AOT
    - » Supersedes Spring Native
- Benefits
  - Shorter startup times
  - No warm-up required
  - Less memory required
  - Fit for scaling up and down
    - » Even to zero

- Spring AOT
  - Creates and inspects an ApplicationContext

public static void main(String[] args) {
 ApplicationContext ctx = SpringApplication.run(ProductServiceApplication.class, args);

- Closed world assumption
  - » Classpath fixed and Spring Beans are defined at build time
  - » Minimize memory footprint
- Generates start-up code
  - » Creates a static ApplicationContext
  - » Programmatic registration of Spring Beans

#### product-service

- build/generated
- ✓ aotSources/se/magnus/microservices/core/product
- J ProductServiceApplication\_\_ApplicationContextInitializer.java
- J ProductServiceApplication\_Autowiring.java
- J ProductServiceApplication\_\_BeanDefinitions.java

Replaces the slow reflection based startup

- Spring AOT
  - Generates native configuration
- Recall from Cadec 2021:
  - GraalVM native-image compiler transforms Java bytecode to an executable image
  - Can't figure out dynamic behavior
    - E.g. use of reflection, dynamic proxies, and local resources
  - Described in a native configuration

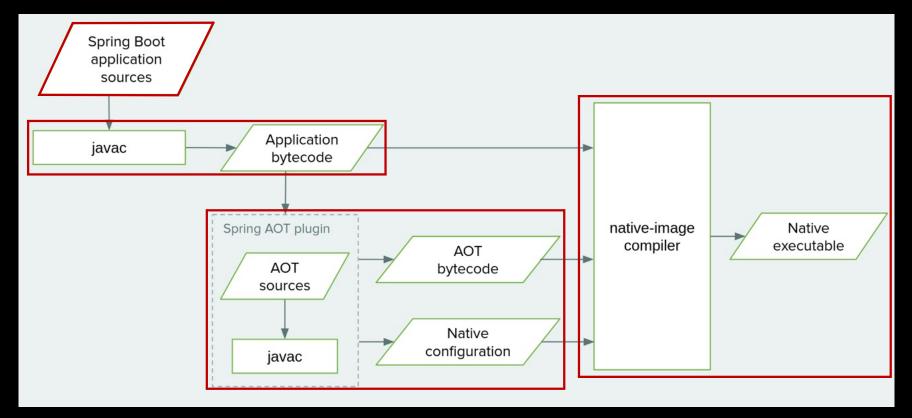


- Spring AOT
  - Generates native configuration
  - Sample of generated native configuration
    - aotResources [aot] resources root
      - META-INF.native-image.se.magnus.microservices.core.product.product-service
        - log native-image.properties
        - {} proxy-config.json
        - {} reflect-config.json
        - {} resource-config.json
        - {} serialization-config.json
  - If Spring AOT fails, we can add native hints
    - » E.g. JSON mapping with Jackson

@RegisterReflectionForBinding({Event.class, Product.class})

» Sample error message:

Error: No serializer found for class se.magnus.util.event.Event This appears to be a native image, in which case you may need to configure reflection



https://spring.io/blog/2021/12/09/new-aot-engine-brings-spring-native-to-the-next-level

- More on Spring AOT
  - AOT tests
    - » Builds a native image and runs tests inside it

#### ./gradlew nativeTest

- » Detects missing Spring Beans and Reflection metadata
- » Best to run in a CI/CD build pipeline
- Use AOT start-up code with Java VM (a.k.a AOT mode)
  - » Shorten startup time in Java VM with ≈20%

java -Dspring.aot.enabled=true -jar app.jar

» Log output

... Starting AOT-processed ProductServiceApplication using Java 17.0.5 ...

- Stability over time
  - Spring ecosystem
    - » Spring AOT smoke tests

Spring-projects / spring-aot-sn	noke-tests
<> Code 🕢 Issues 28 11 Pull requ	uests 3 🕑
१ main ▾ 1 branch ⓒ 0 tags	
wilkinsona Test against Spring Boot 3.0	.2-SNAPSHOT
aot-smoke-test-support	Add test for M
aot-smoke-test-third-party-hints	Remove obsol
batch/batch	Update Batch
boot	Add test for XI
🖿 ci	Add smoke tes
Cloud	Polish "Add Cl
🖿 data	Add test for M
framework	Add webmvc-
💼 gradle	Add table of c
💼 graphql	Use document
integration	Smoke test for
security	Correct assert
session	Remove const

#### - 3PP libraries

#### » GraalVM reachability metadata

🖓 or	acle / graalvm-reachability-metadata
<> c	ode 🕢 Issues 22 🏦 Pull requests 14
۲.	master - graalvm-reachability-metadata / m
۲	dnestoro Merge pull request #148 from dsyer/grpc
	ch.qos.logback/logback-classic
	com.ecwid.consul/consul-api
	com.graphql-java/graphql-java
	com.h2database/h2
	com.mysql/mysql-connector-j
	com.sun.mail/jakarta.mail
	com.zaxxer/HikariCP
	io.grpc/grpc-netty
	io.netty
	io.undertow/undertow-core
	mysql/mysql-connector-java
	net.java.dev.jna/j <b>na</b>
	org.apache.commons/commons-pool2

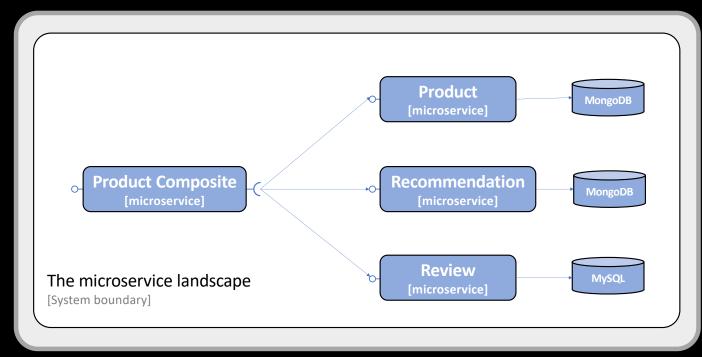
- How?
  - Add GraalVM's build plugin:

plugins {
 id 'org.graalvm.buildtools.native' version '0.9.18'

- Build a native image: ./gradlew nativeCompile
  - » Requires GraalVM JDK and native-image compiler
  - » OS & HW specific, e.g. macOS and ARM64
- Build Docker image: ./gradlew bootBuildImage
  » HW specific, e.g. Intel x86\_64
- Ongoing work
  - <u>GraalVM Cross-compilation support</u>
  - Paketo buildpacks Add support for ARM64
  - Paketo buildpacks 2023 Roadmap
  - Callista blog post Docker images on ARM64

### **TEST:** NATIVE COMPILE

- System landscape from the 2ed of my book
  - Migrated to Spring Boot 3





#### **TEST:** NATIVE COMPILE TIMES

- Compile times with ./gradlew nativeCompile
  - Minimal Spring Boot app

```
Finished generating 'demo' in 32,3s.
[native-image-plugin] Native Image written to: /Users/magnus/
```

- The Product service from the test landscape

```
Finished generating 'product-service' in 1m 41s.
[native-image-plugin] Native Image written to: /Users/magnus/
```

- Not fast enough for a TDD loop, but sufficient for a CI/CD build pipeline
- But significantly better than 2021

native-image --initial
[total]: 687,593.47 ms

### **TEST:** STARTUP TIMES

- Java VM microservices
  - Started ProductServiceApplication in 4.988 seconds
  - Started ProductCompositeServiceApplication in 5.495 seconds
  - Started ReviewServiceApplication in 5.442 seconds
  - Started RecommendationServiceApplication in 4.886 seconds
- Native image microservices
  - Started ProductCompositeServiceApplication in 0.148 seconds
  - Started RecommendationServiceApplication in 0.198 seconds
  - Started ProductServiceApplication in 0.184 seconds
  - Started ReviewServiceApplication in 0.229 seconds

Native image app starts up 25 times faster than JVM app

#### **TEST:** MEMORY USAGE AFTER STARTUP

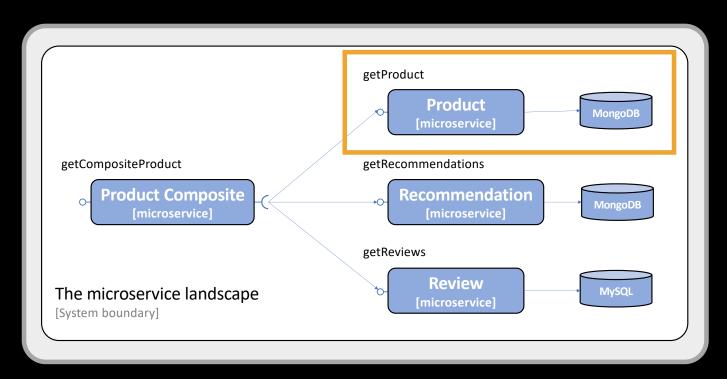
- Java VM microservices
  - review-1 239.2MiB
  - product-composite-1 **216.5MiB**
  - product-1 **212.6MiB**
  - recommendation-1 215.5MiB
- Native image microservices
  - product-1 78.53MiB
  - recommendation-1 78.55MiB
  - product-composite-1 55.84MiB
  - review-1 70.14MiB

Native image app requires less memory to startup.

But what happens over time?

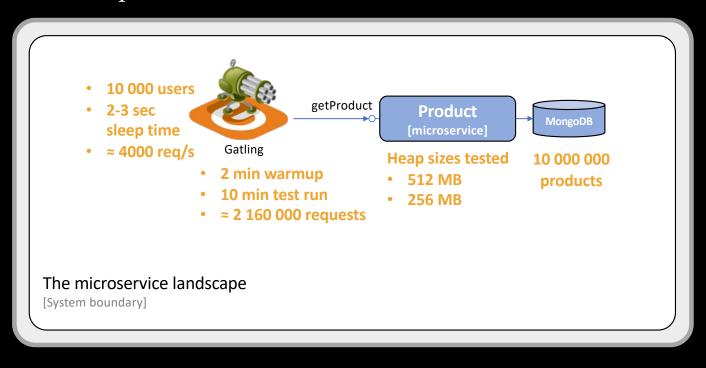
### **TEST:** RESOURCE USAGE OVER TIME

• Test scope



### TEST: RESOURCE USAGE OVER TIME

• Test setup



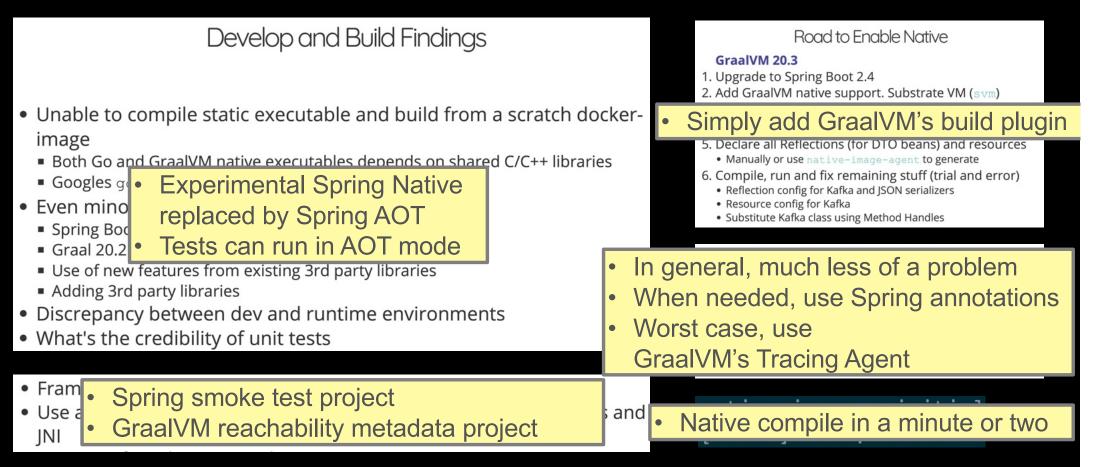
### **TEST:** RESOURCE USAGE OVER TIME

### • Test results

Test case	No of calls		CPU time (min:sec)	Threads	50 % (ms)	95 % (ms)
Native, 256 MB	2 159 056	206	20:54	42	2	11
JVM, 256 MB	2 162 800	235	15:43	54	2	5
Native, 512 MB	2 158 443	220	20:23	42	2	9
JVM, 512 MB	2 163 090	347	15:52	54	2	5



• Reiterate the concerns from Cadec 2021



- Summary
  - Concerns from Cadec 2021 mitigated with Spring Boot 3
  - **Startup**: Native 25 times faster than JVM
  - Memory: Native beats JVM
  - CPU: JVM Hotspot beats native
  - Try it out, if start-up time is important!



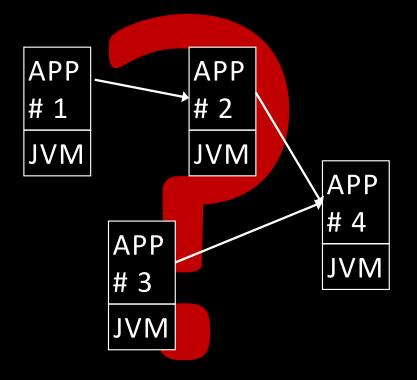


### AGENDA

- Overview
- Migration
- Native Compile
- Observability
- Summary

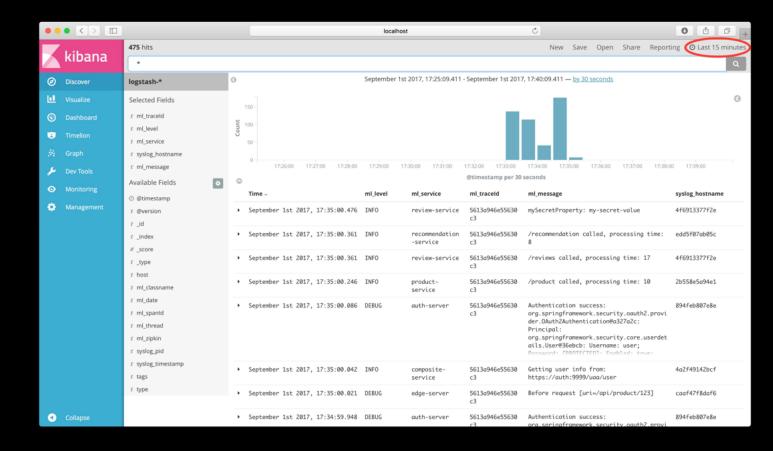


• Observability = Logging + Tracing + Metrics





• Observability = **Logging** + Tracing + Metrics





#### WHAT HAPPENED?

• Observability = Logging + **Tracing** + Metrics

🗧 😑 🔹 🔹 Kiali Console	× +					
			s=hands-on&start=1572775376329000&e		klookback=3600&maxDura 📩	a 🦊   🌑 O
= 🔍 kiali				XX		∂ admin -
	Namespace: hands-on ~					
Overview	Service * product-o	omposite 👻	Lookback * La	ist 1h		Traces
Graph	> Show Advanced Options					
Applications	← ✓ istio-ingress product-con	gateway: product-co posite* 269ab71	mposite.hands-on.svc.clu	ster.local:80/	earch	
Workloads	Trace Start November 3, 2019 12:0	1 PM   Duration 38.07ms   Services	6 Depth 4 Total Spans 11			00.07
Services	Oms	9.52ms	19.04ms	_	28.55ms	38.07ms
Istio Config	Service & Operation	✓ > ⊗ ≫0ms	9.52ms	19.04ms	28.55ms	38.07ms
Distributed Tracing	<ul> <li>istio-ingressgateway product-co</li> <li>istio-ingressgateway (product-composite.hands-product-composite.hand</li></ul>	istio-mixe	1ms			
	product-composite.hands-on product.hands-on product.h				15.16ms	
	product-composite.hands-on     recommendation.hands-on     product-composite.hands-on	recommendat	23.82ms		16.01ms	_
	review.hands-on review.hand		23.3ms			



#### PROCESSING TIME?

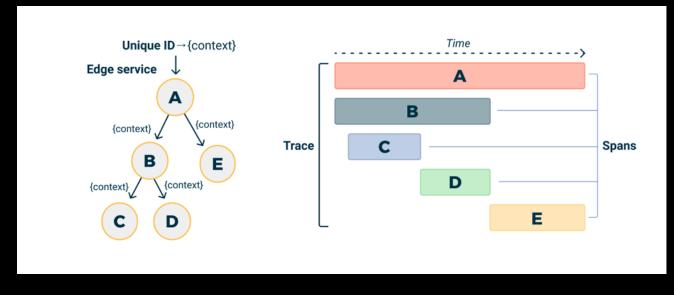
• Observability = Logging + Tracing + Metrics



# $\begin{array}{c} APP \\ \# 1 \\ \end{array} \xrightarrow{} APP \\ \# 2 \\ \end{array} \xrightarrow{} APP \\ \# 3 \\ \end{array}$

#### **RESOURCE USAGE?**

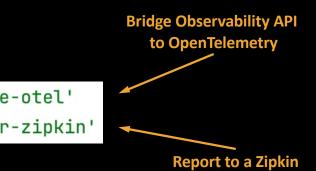
- Observability in Spring Framework 6.0
  - Logs and Metrics already supported in 5.0
  - Tracing: New module, Micrometer Tracing
    - » Based on spring-cloud-sleuth
    - » Traces are reported as a trace tree of spans based on OpenTelemetry
    - » Contexts based on W3C Trace Context



- Tracing in Spring Framework 6.0
  - Built-in support
    - » Creates traces for incoming requests, if missing
    - » Propagates to outgoing requests
    - » Supports both synchronous and asynchronous requests
    - » Propagates to logs
  - Dependencies

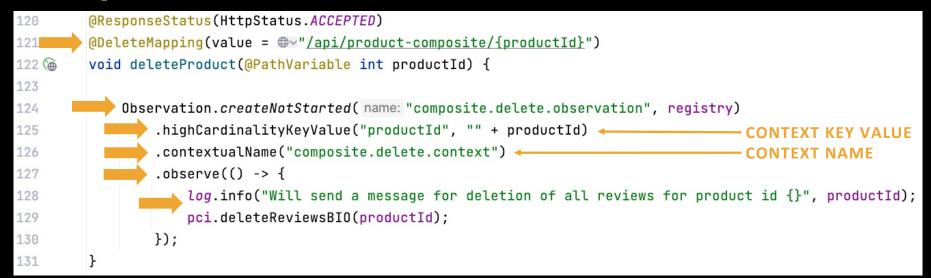
implementation 'io.micrometer:micrometer-tracing-bridge-otel'
implementation 'io.opentelemetry:opentelemetry-exporter-zipkin'

- » Support for alternative <u>Tracer Implementations</u>
- No auto propagation (yet) for reactive libraries, e.g. Spring WebFlux
  - » Spring Boot 3 Webflux project missing traceId and spanId in logs
  - » Context Propagation Library

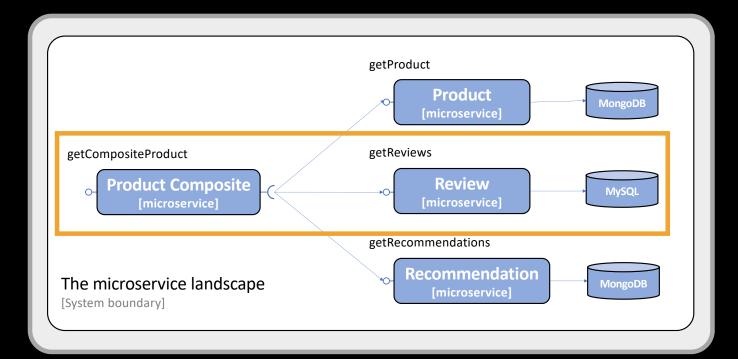


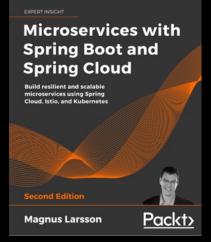
compatible tracer

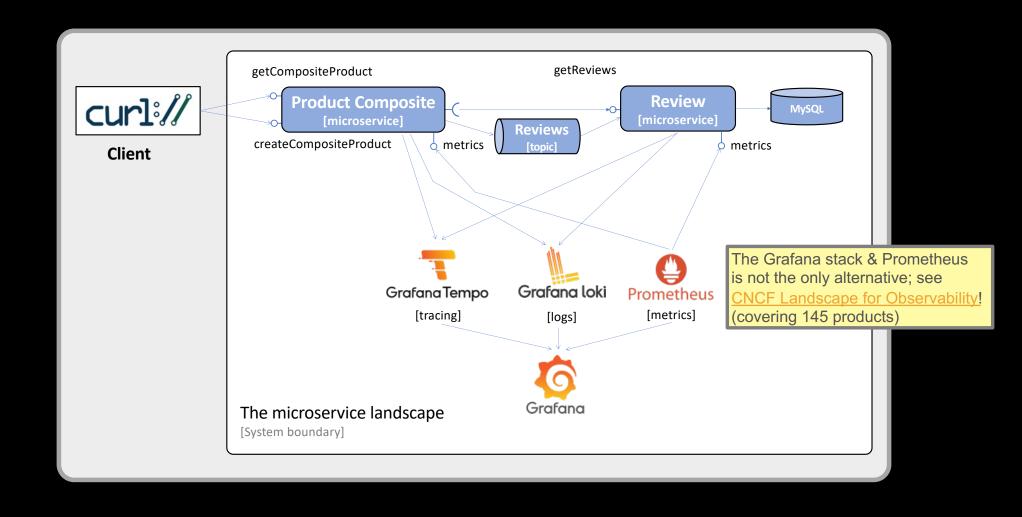
- Tracing in Spring Framework 6.0
  - Programmatically
    - » Spring abstraction **Observation**
    - » Custom spans and contexts can be created









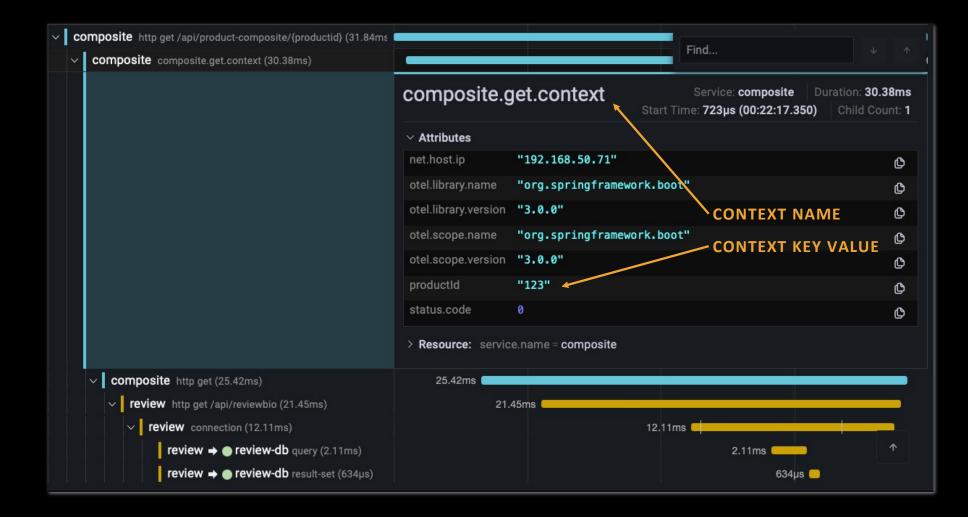


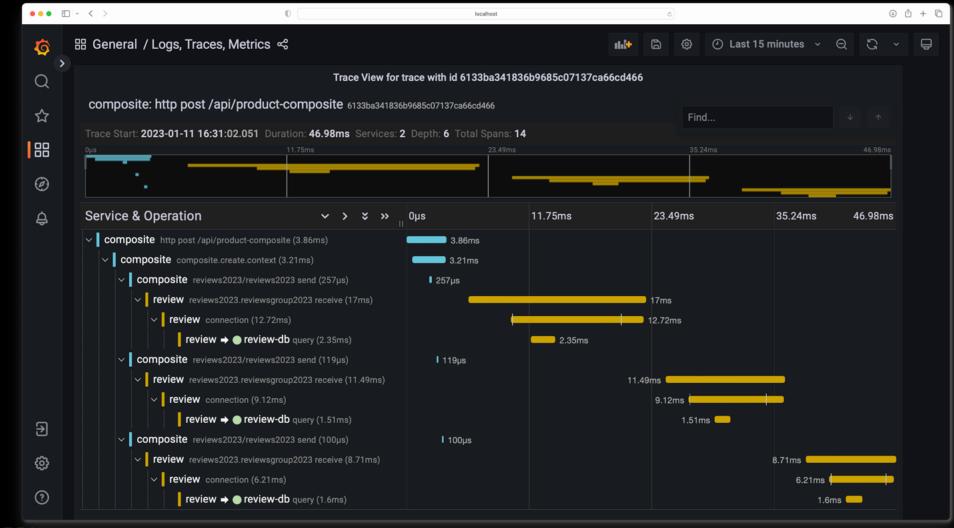
	< >	0	loci	alhost		Ċ				$( \mathbf{J} )$	≏ +	0
🖸 M 🛆			🧔 Logs,	Traces, Metrics - Dashboard	ds - Grafana							_
🌀 🔠 Gen	eral / Logs, Traces, Met	trics 🗠			nhi+		) 🕘 Last	5 minutes		S		Ð
Trace ID	58131f06fe0424e9099eafc	d0a1be2580 ~										
		Logs w	vith trace ID 58131f0	)6fe0424e9099eafd0	a1be2580							
슈   > 2022	2-12-31 00:22:17 composite	2022-12-31T00:22:17.	350+01:00 INFO [	[composite,58131f6	∂6fe0424e9	099eafd0	a1be2580,3ef	ac4cd0a394	40d1] 7:	378		
B   2022	2-12-31 00:22:17 review	http-nio-6543-exec-7 2022-12-31T00:22:17.										
) 2022	2-12-31 00:22:17 review	p-nio-7654-exec-7] c 2022-12-31T00:22:17.									tt	
۵		p-nio-7654-exec-7] o roduct_id,r1_0.revie		t,r1_0.version fro			_0.id,r1_0.a ere r1_0.pro		.conte	nt,r1_0		
⇒ ° 2022	2-12-31 00:22:17 review	2022-12-31T00:22:17. p-nio-7654-exec-7] c								[h	tt	
> 2022	2-12-31 00:22:17 composite	2022-12-31T00:22:17. http-nio-6543-exec-7				099eafd0 : Got 3		ac4cd0a394	40d1] 7:	378		
Common	labels: MagnusMBP 58131f06fe042											
		Trace View	for trace with id 581	31f06fe0424e9099e	afd0a1be2!	580						
com	posite: http get /api/pro	duct-composite//pro	ductid construction	0404-0000640-16-00	:00							
CON		duct-composite/{pic	Juucius seisitusie	0424e9099eatd0a1be25	580							
						r	ind					
	Start: 2022-12-31 00:22:17.34	19 Duration: <b>31.84ms</b> Se 7.96ms	rvices: 2 Depth: 6 T	Total Spans: 7			23.88ms					
Trace S	Start: 2022-12-31 00:22:17.34	9 Duration: 31.84ms Se 7.96ms	rvices: 2 Depth: 6 T	Total Spans: <b>7</b> 15.92ma						ф 1 31.84	ims	
Trace S	Start: 2022-12-31 00:22:17.34	99 Duration: 31.84ms Se 2.96ms	rvices: 2 Depth: 6 T	Total Spans: 7						ф ? 31,84	ims	
Oµs	start: 2022-12-31 00:22:17.34	9 Duration: 31.84ms Se 7.96ma ~ > ~ > ~ >		Total Spans: 7		15.92	23.88ms	23.88	Bms	31.84 31.84	ms	
Servia	_	7.96ms	0µs	15.92ms			23.88ms	23.88	ßms	31.84 31.84	ms	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-c composite composite.get.com	2.96ms	Uus	15.02ms 7.96ms			23.88ms	23.88	ßms	31.84 31.84	ms 31	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get.com v composite http get (25.42	2.96ms	0µs	15.92ma			23.88ms	23.88	ßms	31.84 31.84	ms 3i	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get.con v composite http get (25.42 v review http get /api/re	2.96ms ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ >	Uus	15.02ms 7.96ms		15.92	23 Billions	23.86	đms	31.84	ms 3	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get con v composite http get (25.42 v review http get /api/re v review connection	2.96ms ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ > ~ >	Uus	15.92ma			23.88ms	23.88	Brms	31.84 31.84	ms	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get con v composite http get (25.42 v review http get /api/re v review connection	2.96ms Composite/(productid) (31.84 text (30.38ms) 2ms) viewbio (21.45ms) (12.11ms)	<mark>1)</mark> Оµs 25.42ms	15 92ms 7.96ms 21.45ms		15.92	23.88ms		dms	31.84	ms 31	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get con v composite http get (25.42 v review http get /api/re v review connection	2.96ms Composite/(productid) (31.84 text (30.38ms) 2ms) viewbio (21.45ms) (12.11ms)	<mark>1)</mark> Оµs 25.42ms	15.92ma		15.92	23.88ms		Brms	31.84	ms	
Cµs Servia ✓ Cou	ce & Operation mposite http get /api/product-o composite composite get con v composite http get (25.42 v review http get /api/re v review connection	2.96ms Composite/(productid) (31.84 text (30.38ms) 2ms) viewbio (21.45ms) (12.11ms)	<mark>1)</mark> Оµs 25.42ms	15 92ms 7.96ms 21.45ms		15.92	23.88ms		Rms	31.84	ms 3	
Servic v   co v	ce & Operation mposite http get /api/product-o composite composite get con v composite http get (25.42 v review http get /api/re v review connection	2.96ms Composite/(productid) (31.84 text (30.38ms) 2ms) viewbio (21.45ms) (12.11ms)	<mark>1)</mark> Оµs 25.42ms	15 92ms 7.96ms 21.45ms		15.92	23.88ms		ims	31.84	ms 31	
Qu ms 20 ms 10 ms = (3)	ce & Operation mposite http get /apl/product- composite composite get con composite http get (25.43       composite http get /apl/re       review http get /apl/re       review ormection       review → ● re       00-18:00 00:18	7.56ms Composite/(productid) (31.84 itext (30.38ms) 2ms) viewbio (21.45ms) i(12.11ms) eview-db query (2.11ms)	III Oµs Imt 25.42ms Iaten	15 92ms 7.96ms 21.45ms	30 0	15.92	23.88ms			31.84 31.84	31	
Qu ms 20 ms 10 ms = (3)	ce & Operation mposite http get /api/product- composite composite.get.com > composite http get /25.42 > review http get /api/re > review connection review + • re	7.56ms Composite/(productid) (31.84 itext (30.38ms) 2ms) viewbio (21.45ms) i(12.11ms) eview-db query (2.11ms)	III Oµs Imt 25.42ms Iaten	15 92ms 7.96ms 21.45ms	30 0	15.92 12.11ms	22.38ms			• •	31	

# LOGS

# TRACES

# METRICS





### AGENDA

- Overview
- Migration
- Native Compile
- Observability
- Summary



### SUMMARY

- With Spring Boot 3, a new foundation is in place
  - Expect a lot of improvements to come over the following years...
- Migration
  - Upgrade to Java 17 and jakarta package names
  - Remove deprecated code
- Native Compile
  - Use if start-up time is important
  - Test and build native images in CI/CD build pipeline
  - Reduce startup times running Java VM in AOT-mode
- Observability
  - Built-in auto-configuration for tracing
  - One interface, **Observation**, to abstract them all
  - One dashboard to observe them all

### QUESTIONS?





#### ML@CALLISTAENTERPRISE.SE



### MAGNUSLARSSONCALLISTA

EXPERT INSIGHT

# Microservices with Spring Boot and Spring Cloud

Build resilient and scalable microservices using Spring Cloud, Istio, and Kubernetes

LEGACY WARNING BASED ON SPRING-BOOT2

**Second Edition** 

**Magnus Larsson** 

Packt>

### QUESTIONS?





#### ML@CALLISTAENTERPRISE.SE



#### MAGNUSLARSSONCALLISTA

EXPERT INSIGHT

# Microservices with Spring Boot 3.0 Spring Cloud

Build resilient and scalable microservices using Spring Cloud. Istio. and Kubernetes

> Mid-year 2023 (hopefully...)





**Magnus Larsson** 

Packt>